The report on porting and running CY32T1 on SGI Altix in Croatia

Summary: The code was compiled using *gmkpack* with optimisation level 2. Compilation errors were removed from *suphy0* and *rad1cnne*. Runtime error in *susta* (SIGFPE) disappeared when printout was introduced. Two more SIGFPE runtime errors in *acveg* and *actqsat* were removed when routines were compiled with optimization level 1. The code is running for all configurations used in Zagreb (927 and 001 for Aladin and Alaro, it was also tested for Aladin with MF physics set-up) but producing different norms and having very poor cputime per time-step performance when new gridpoint 3D variables are used.

Introduction

The code is ported to SGI Altix (24 Intel Itanium2 CPUS, SUSE Linux enterprise server 9.0, Intel compiler for Fortran and C++) in Zagreb using *gmkpack*. Optimization level 2 was used (3 being most optimized and 0 the least).

Two routines (*suphy0* and *rad1cnne*) were modified due to compilation errors (extra comma at the end of format specification statement) and an executable was obtained. Also several routines had to be compiled with lower level of optimisation to avoid runtime abort due to SIGFPE. In CY29T2 it was accvimp and accvimpd, but now these two did not pose any problem (identified so far).

The code is able to run e927 and e001 and complete 72 hour forecast. There were no problems with the "plain" Aladin test (without prognostic TKE, cloud and precipitation species). But, when microphysics was used, serious problems with cputime per timestep appeared, even worse than in CY29T2 with Alaro modifications set.

Runtime execution problems

1. **Abort due to SIGFPE**

a) susta

Both 001 and e927 were aborting in susta due to SIGFPE in line 183. A simple printout helped fix the problem, and there was no SIGFPE in that line (probably since compiler interpreted the code differently). Since susta did not change between CY29T2 (where this problem did not occur) and CY32T1, this is very strange.

b) acveg and actqsat

001 was aborting in one and afterwards in the other routine with SIGFPE, both were compiled with the lower level of optimization and the model run afterwards. It is difficult to say where and why the error occurred and which piece of code was optimized in the wrong way since the model printout did not give the exact line of SIGFPE this time.

There is a remaining concern on how much of the other code was misinterpreted by the compiler due to optimization. The older versions of Aladin that were compiled (the whole package) with different optimizations had produced significantly different norms (differences already after a few digits).

2. Cputime per time-step problem

The cputime per timestep varies during integration as well as it does with CY29T2 when new 3D gridpoint variables are introduced for pTKE and microphysics. In CY29T2 with Alaro0 the cputime per time-step follows the same general pattern: it grows during forward DFI, reaches some maximum value either during forward DFI or during the first 6 hours of forecast and then decreases to the value from the beginning of forward DFI. Runs from different analyses reach different maximum cputime per timestep at different forecast step, but when the same run (for the same date) is repeated the pattern is the same, even with different number of processors.

In CY32T1 cputime per timestep grows during forward DFI, drops to the normal value (from the beginning of the forward part of DFI) at the beginning of the forecast run, but then it grows again and significantly varies during the forecast run. The complete forecast run time is by far too long to be acceptable for operational purpose.

It is possible to see the memory and CPU usage for each CPU on the computer. It showed that during the timesteps with normal cputime per timestep usage, it is only the "user" that is using CPU, but after a while during the run in the forward DFI or whenever cputime per timestep increases, it can be seen that the operating system is using the same cpus, and doing so quite severely. This does not happen when additional variables used in microphysics are not used.

a) Testing on hpce

Since CY32T1 was ported to **hpce,** the same tests were performed on it, but using 32 processors and NPROMA=30. There were no problems with cputime per time-step. It kept the same value during the whole forecast run. Up to our knowledge, no other service (even those using SGI Altix) has a similar experience to ours.

b) LIMP, LIMP_NOOLAP ... and LSLONDEM

None of the options in NAMPAR0 and NAPAR1 is the cause of the problem, switching them off only increases the overall cputime. The largest increase in profiling cputime is observed for trltom, trgotl, laitri, laitli_hd, slcomm and similar routines that require communication between processors. Further tests, with different NPROMA values (since very large NPROMA reduces the amount of communication between processors).

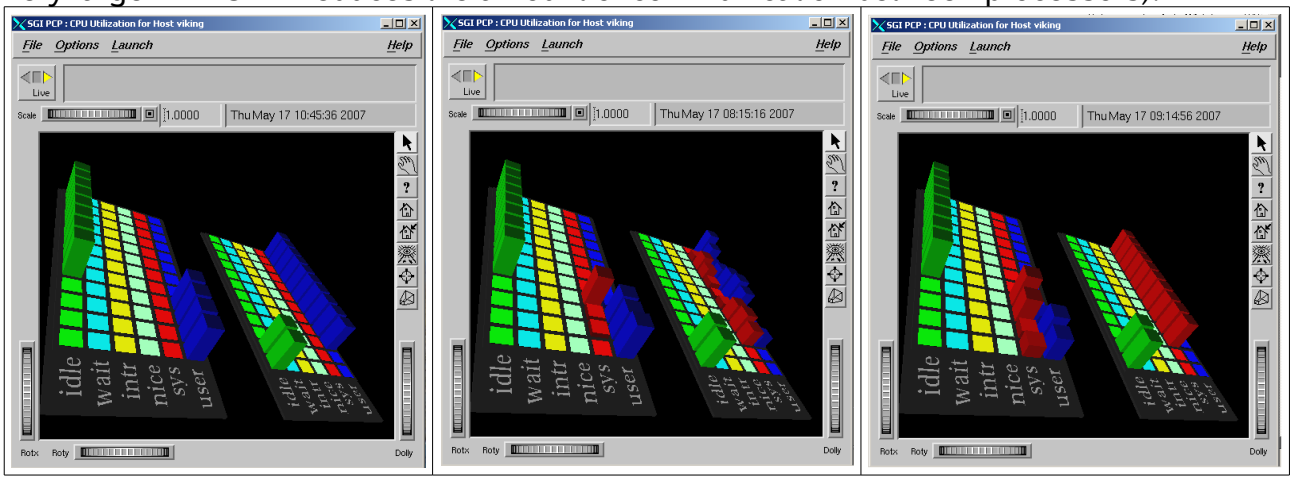

Figure 1. CPU usage for each CPU on SGI Altix in Croatia for normal run (left), slight increase of cputime per time step (center) and severe increase (right) during an Alaro run with CY32T1. Green columns show that processors are idle, red columns that CPU is being used by operating system and blue that CPU is being used by an user (aladinhr).

c) NPROMA

The operational NPROMA used with CY29T2 (for Aladin and Alaro) is 80. The details of cputime per timestep for AL29T1 are in Table 1.

Table 1: Cputime per timestep for operational Aladin and Alaro with CY29T2 and NPROMA=80 on 14 procesors during backward DFI (adiabatic) and otherwise, forward DFI and forecast (with physics).

	adiabatic	***with physics***
Aladin	0.82	2.35
Alaro	0.88	3.23

Increased cputime per timestep between CY29t2 and CY32t1 even for a plain Aladin run without any microphsics variables has encouraged testing of different NPROMA values both for the basic configuration (called Aladin) and an Alaro0 configuration.

NPROMA was varied form 30 to 3360 (maximum allowed for the Croatian domain on 14 CPUs). Minimum cputime per timestep is found for the largest NPROMA allowed on 14 processors (3360) as can be seen in Table 2. As far as our knowledge and understanding of the computer architecture is reaching, we are supposed to be using SGI Altix with Intel Itanium2 processors that are supposed to be scalar. However this result is a feature of vector machines.

Table 2: Cputime per timestep for operational Aladin and Alaro with CY32T1 for different NPROMA values on 14 processors during backward DFI (adiabatic) and otherwise, forward DFI and forecast (with physics).

	adiabatic	***with physics***
Aladin NPROMA=30	5.25	9.27
Aladin NPROMA=80	2.3	4.80
Aladin NPROMA=336	1.1	3.15
Aladin NPROMA=3360	0.95	3.05
Alaro NPROMA=30	7.33	21.53
Alaro NPROMA=80	3.35	9.60
Alaro NPROMA=336	1.7	4.85
Alaro NPROMA=3360	1.17	3.75

Conclusion

It is possible that some of the routines that communicate between processors are interpreted in a wrong way by the compiler. Since the problems observed on SGI Altix in Croatia were not reproduced on any other computer (NECs in Prague and Toulouse as well as VPP in Meteo France and hpce IBM in Reading), it can be only concluded that the problem is in the version of the compiler used here or the operating system.

Martina Tudor and Stjepan Ivatek-Šahdan